

Remarks

Specification

The Office has objected to the abstract, requiring that it be limited to a single paragraph. Applicants have amended the abstract to conform to the required format. Applicants respectfully request that the Examiner withdraw the objection. Such action is respectfully requested.

Claims Objections

The Office has asserted an objection of claims 1-32 because of the abbreviation of the word conformance-test (“CT”). Applicants have amended the claims to remove the abbreviation. Applicants respectfully request that the Examiner withdraw the objection. Such action is respectfully requested.

Statutory Subject Matter

The Office has asserted a rejection of claims 1-11 as directed at non-statutory subject matter. As suggested by the Examiner, Applicants have amended claims 1-11 to recite “computer implemented” methods. Claims 1-11 are now in condition for allowance. Such action is respectfully requested.

Patentability over Leiba

The Office has asserted a rejection of claims 1-3, 5, 7-14, 16, 18-25, 27, and 29-32 under 35 USC §102(e) as anticipated by Leiba, U.S. Patent No. 5,883,661 (“Leiba”). Applicants respectfully traverse.

Claim 1

Applicants respectfully submit that Leiba fails to anticipate amended claim 1, because Leiba fails to teach or suggest “applying the software implementation and the software specification to produce a conformance-test enabled implementation comprising portions of the software implementation and the software specification integrated into a same body of code.” Specifically, amended claim 1 recites,

1. (currently amended) A computer-implemented method of conformance-testing a software implementation with a software specification, the method comprising:

applying the software implementation and the software specification to produce a conformance-test enabled implementation comprising portions of the software implementation and the software specification integrated into a same body of code wherein nondeterministic choices of the software specification result in assigning a corresponding choice of the conformance-test enabled implementation to a variable; and the conformance-test enabled implementation comprising a test that the variable comprises one of the nondeterministic choices of the software specification.

For example, the Application states,

Once the specification and implementation are compiled to a common IL, portions of code from both may be applied to produce the conformance-test enabled implementation code. Producing and executing a single body of code alleviates difficulties that arise from separately executing the implementation IL and the specification IL and then attempting to compare the results of the separate executions. (*Page 6, lines 7-11*)

Further, Figure 1 of the Application illustrates that portions of code from both the implementation and the specification are integrated into a single body of code. Applicants respectfully submit that Leiba fails to anticipate amended claim 1, because Leiba fails to teach or suggest “applying the software implementation and the software specification to produce a conformance-test enabled implementation comprising portions of the software implementation and the software specification integrated into a same body of code.”

For example, the Office asserts the following passages against the recited arrangement:

The present invention also includes a method to verify the compliance standards of a software application.

(*Col. 3, lines 62-63*)

... receiving at least one response from the server application and comparing the at least one response with expected responses for performing compliance testing based on the response requirements.

(*Col. 2, lines 11-12.*)

... each response including at least one component and the method may further include the steps of permitting non-deterministic ordering of the responses and permitting ordering of components within each response.

(*Col. 2, lines 28-31*)

The analyzer component can be instructed to allow for permissible non-deterministic ordering of responses and ordering of components within any given response.
(Col. 4, lines 7-10)

As understood by Applicants, Leiba teaches directly away from “applying the software implementation and the software specification to produce a conformance-test enabled implementation comprising portions of the software implementation and the software specification integrated into a same body of code.”

For example, Leiba indicates that a “test processor component of a server conformance tester ... issues one or more command sequences to a protocol server, and receives one or more responses from the server” (col. 4, lines 1-4). The sequences are issued over a “pipe 219 which is a communication component that establishes a connection between the conformance tester 130 and the server 140” (col. 6, lines 11-13). This is similar to black box testing, where commands are issued on an implementation externally via an API, and the responses are received from the implementation and tested against expected data. This analysis is further supported by the following discussion in Leiba regarding Figure 1.

FIG. 1 depicts a high-level design of the conformance test environment, where client test data 100 is processed by the test engine 130, then transmitted to a server 140. The server 140 responses are then processed to create a test result 150.
(Col. 4, lines 50-54)

Leiba is describing a prior art version similar to black box testing, where the test code and/or environment is separate from the implementation under test. For at least this reason, Leiba fails to teach or suggest “applying the software implementation and the software specification to produce a conformance-test enabled implementation comprising portions of the software implementation and the software specification integrated into a same body of code.”

For at least this reason, claim 1 is allowable. Such action is respectfully requested.

Claims 2-6

Claims 2-6 depend from claim 1. Since they depend from claim 1, they should be allowed for at least the reasons stated for claim 1. In view of the foregoing discussion of claim 1, the merits of the separate patentability of dependent claims 2-6 are not belabored at this time. Claims 2-6 should be allowable. Such action is respectfully requested.

Independent Claims 7, 12, 18, 23, 29

Applicants respectfully submit that for reasons similar to those stated above, such as for claim 1, Leiba fails to anticipate the following features:

Claim 7 — “producing a software object organized such that a step of the software specification and a corresponding code section of the software implementations are integrated in the software object”

Claim 3 — “a same body of code with portions from both the software implementation and the software specification”

Claim 18 — “producing a software object organized such that a step of the software specification and a corresponding code section of the software implementation are integrated in the software object”

Claim 23 — “a same body of code with portions from both the software implementation and the software specification”

Claim 29 — “producing a software object organized such that a step of the software specification and a corresponding code section of the software implementation are integrated in the software object”

Since Leiba fails to anticipate these features of independent claims 7, 12, 18, 23, and 29, they should be allowable. Such action is respectfully requested.

Dependent Claims 8-11, 13-17, 19-22, 24-28, and 30-32

Claims 8-11, 13-17, 19-22, 24-28, and 30-32 depend from the above allowable independent claims. Since claims 8-11, 13-17, 19-22, 24-28, and 30-32 depend from the above allowable independent claims, they should be allowed for at least the above reasons. Such action is respectfully requested.

Patentability Over Leiba in view of Willis

The Office has asserted a rejection of claims 4, 6, 15, 17, 26, and 28 under 35 USC §103(a) as obvious over Leiba in view of Willis, U.S. Patent No. 6,321,376 (“Willis”). Applicants respectfully traverse.

Claims 4 and 6

Claims 4 and 6 depend from claim 1. Since claim 1 is allowable over a Leiba-Willis combination, claims 4 and 6 must also be allowable. For example, Applicants respectfully submit, that the proposed Leiba-Willis combination fails to teach or suggest “applying the software implementation and the software specification to produce a conformance-test enabled implementation comprising portions of the software implementation and the software specification integrated into a same body of code.”

For example, the Application states,

Once the specification and implementation are compiled to a common IL, portions of code from both may be applied to produce the conformance-test enabled implementation code. Producing and executing a single body of code alleviates difficulties that arise from separately executing the implementation IL and the specification IL and then attempting to compare the results of the separate executions. (*Page 6 lines 7-11*)

Further, Figure 1 of the Application illustrates that portions of code from both the implementation and the specification are integrated into a single body of code. Applicants respectfully submit that Leiba fails to anticipate amended claim 1, because Leiba fails to teach or suggest “applying the software implementation and the software specification to produce a conformance-test enabled implementation comprising portions of the software implementation and the software specification integrated into a same body of code.”

For example, the Office asserts that the following Leiba passages against the recited arrangement:

The present invention also includes a method to verify the compliance standards of a software application.

(*Col. 3, lines 62-63*)

... receiving at least one response from the server application and comparing the at least one response with expected responses for performing compliance testing based on the response requirements.

(*Col. 2, lines 11-12.*)

... each response including at least one component and the method may further include the steps of permitting non-deterministic ordering of the responses and permitting ordering of components within each response.

(Col. 2, lines 28-31)

The analyzer component can be instructed to allow for permissible non-deterministic ordering of responses and ordering of components within any given response.

(Col. 4, lines 7-10)

As understood by Applicants, Leiba teaches directly away from “applying the software implementation and the software specification to produce a conformance-test enabled implementation comprising portions of the software implementation and the software specification integrated into a same body of code.”

For example, Leiba indicates that a “test processor component of a server conformance tester ... issues one or more command sequences to a protocol server, and receives one or more responses from the server” (col. 4, lines 1-4). The sequences are issued over a “pipe 219 which is a communication component that establishes a connection between the conformance tester 130 and the server 140” (col. 6, lines 11-13). This is similar to black box testing, where commands are issued on an implementation externally via an API, and the responses are received from the implementation and tested against expected data. This analysis is further supported by the following discussion in Leiba regarding Figure 1.

FIG. 1 depicts a high-level design of the conformance test environment, where client test data 100 is processed by the test engine 130, then transmitted to a server 140. The server 140 responses are then processed to create a test result 150.

(Col. 4, lines 50-54)

Leiba is describing a prior art version similar to black box testing, where the test code and/or environment is separate from the implementation under test. For at least this reason, Leiba fails to teach or suggest “applying the software implementation and the software specification to produce a conformance-test enabled implementation comprising portions of the software implementation and the software specification integrated into a same body of code.”

Next, the Examiner directs Applicants to the following passage in Willis,

FIG. 4 represents the compiled equivalent of the interpreter embodiment described above. In this embodiment, the formal language specification is parsed into a specification intermediate

(Block 31), then a test case compiler creates a compiled test case generator, potentially using an intermediate programming language manifestation of the generator and associated programming language compiler (both within Block 50) to yield an executable generator. *Col. 7, lines 6-67.*

The resulting compiled generator (Block 51) produces test cases in a manner functionally equivalent to the interpretative generator (Block 32) discussed above in the context of FIG. 3. *Col. 8, lines 1-3.*

In the current art, a single test case may be automatically permuted by character replacement in order to yield a sequence of closely related test cases. For example, a test case may be permuted to write various types and/or values into a file. Such automatic permutation spans a small space within the set of desired tests; generally a single set of syntactic productions and semantic constraints (common to all permuted tests).

Current state of the art in manual test suite development or automatically permuted test cases results in sub-optimal conformance testing fidelity of a tool under test, high development cost and high maintenance cost. An apparatus and means achieving higher fidelity conformance testing with lower development and maintenance effort, as disclosed in the present invention, is novel and useful. *Col. 2, lines 38-53.*

The recited passages in Willis fail to teach or suggest “applying the software implementation and the software specification to produce a conformance-test enabled implementation comprising portions of the software implementation and the software specification integrated into a same body of code.”

Rather, Willis describes a black box testing arrangement similar to that described in Leiba. For example, Willis at Figure 1, item 7, and at Figure 2, item 8, depicts a testing environment separate from the tool under test. Further, at Figure 1, Willis describes a series of generated test sequences at item 12 and item 15, and the corresponding tool responses at items 9, 15, and 16. Thus, Willis teaches directly away from the arrangement recited in claim 1. Further, Willis describes,

The method of test case generation is shown in FIG. 3 (for the interpreter embodiment) and FIG. 4 (for the compiler embodiment). Both the interpreter embodiment (Block 5) and the compiler embodiment (Blocks 3 and 4) utilize the same generator-oriented formal language specification (Block 1), produce the same

stream of output test cases for the tool under test (lumped into Arrows 12) and accept the same execution result feedback (Arrow 9). *Col. 6, lines 53-60.*

Willis describes a "stream of output test cases" and a corresponding "execution result feedback." Thus, Willis teaches directly away from "applying the software implementation and the software specification to produce a conformance-test enabled implementation comprising portions of the software implementation and the software specification integrated into a same body of code."

To establish a prima facie case of obviousness, the Office must direct applicants to references that teach or suggest all of a claim's limitations. Applicants respectfully submit that the Office has failed to carry the burden of establishing obviousness, because the cited art of record fails to teach or suggest "applying the software implementation and the software specification to produce a conformance-test enabled implementation comprising portions of the software implementation and the software specification integrated into a same body of code."

For at least this reason amended claim 1 is in condition for allowance. Since claims 4 and 6 depend from claim 1, they should be allowable for at least the reasons state for claim1. Such action is respectfully requested.

Claims 15, 17, 26, and 28

Applicants respectfully submit that for reasons similar to those stated above in claim 1, the Leiba-Willis combination fails to teach or suggest the following features of claims 12 and 23.

Claim 12 — "a same body of code with portions from both the software implementation and the software specification"

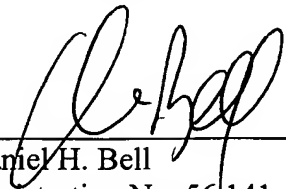
Claim 23 — "a same body of code with portions from both the software implementation and the software specification.

Since claims a Leiba-Willis combination fails to teach or suggest the recited features, claims 12 and 23 should be allowable. Further, claims 15 and 17 depend from claim 12, and claims 26 and 28 depend from claim 23. Since claims 15, 17, 26, and 28 depend from the above allowable independent claims, they should be allowed for at least the above reasons. Such action is respectfully requested.

Respectfully submitted,

KLARQUIST SPARKMAN, LLP

By



Daniel H. Bell
Registration No. 56,141

One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Telephone: (503) 595-5300
Facsimile: (503) 228-9446